AN EVENT DISTRIBUTION PLATFORM FOR RECOMMENDING CULTURAL ACTIVITIES

Toon De Pessemier, Sam Coppens, Erik Mannens, Simon Dooms, Luc Martens
IBBT, Ghent University, Gaston Crommenlaan 8, Ghent, Belgium
{*toon.depessemier, sam.coppens, erik.mannens, simon.dooms, Luc1.martens*}*@ugent.be*

Kristof Geebelen
IBBT, K.U. Leuven, Celestijnenlaan 200A, Leuven, Belgium
kristof.geebelen@cs.kuleuven.be

Keywords: Event recommendation, information filtering, content-based recommendation, service-oriented architecture

Abstract: Today, people have limited leisure time which they want to fill in according to their interests. At the same time, cultural organisations offer an enormous amount of activities via their websites. This scarcity of time and the abundance of cultural events reinforce the necessity of recommender systems that assist end-users in discovering events which they are likely to enjoy. However, traditional recommender systems can not cope with event-specific restrictions such as the availability, time and location of cultural activities. Moreover, aggregating the events, collecting consistent metadata, and enriching these metadata with cross-domain knowledge pose additional challenges for the conventional distribution and recommender systems. In this paper, we show how personalised recommendation, content-based filtering, and distribution of events can be enabled by the enrichment of events metadata via open linked data sets available on the web of data. For consistency across several events providers, we propose an event model using an RDF/OWL representation of the EventsML-G2 standard. Integrating these various functionalities as an extendable bus architecture provides an open, user-friendly event distribution platform that offers the end-user a tool to access useful event information that goes beyond basic information retrieval.

1 INTRODUCTION

Nowadays, people tend to consult interactive and dynamic information from the Internet, rather than static hard-copy press. This shift applies to the cultural scene as well: cultural organisations provide detailed websites which may help to decide how to spend leisure time optimally. Furthermore, the role of these cultural websites is evolving from merely information provider to a guide that puts structure and emphasis on the demand and supply of cultural events and their accompanying assets. To deal with the vast and complex structure of cultural events and leisure activities, we believe that an event model should be used as the fundamental basis in organizing and accessing cultural event information. To handle the overload of information, a recommender system based on this event model is essential to assist users in finding relevant activities.

Our ideas are demonstrated in the context of the end-to-end event distribution platform CUPID[1] which is implemented for the Flemish cultural scene. Using multiple information sources, the events are aggregated and enriched with additional knowledge from large linked data sets. The metadata of these events, described by the EventsML-G2 standard, together with the enriching multimedia assets enable content-based filtering, recommendation and distribution of the events.

The remainder of this paper is organized as follows: Section 2 elaborates on aggregating the events from various decentralized parties and the enrichment based on available knowledge sources. Section 3 describes the utilized EventsML-G2 (International Press Telecommunications Council, 2009) standard completed with its conceptualisation in an OWL (McGuinness and van Harmelen, 2004) ontology as a unifying (meta)data model. This event description is used in the recommender module, which is discussed in Section 4, to generate personal sug-

[1] https://projects.ibbt.be/cupid

gestions. In Section 5 we present the workflow-based bus-infrastructure that is employed to integrate the different functionalities to a continuous flow of cultural events. Finally, we offer a brief conclusion on our research work in Section 6.

2 AGGREGATION & ENRICHMENT

In a first phase, event aggregation is performed; i.e. the retrieval of primary information of cultural activities from different event organisers. This primary information is defined as the minimal information that is necessary to identify and locate the event. In a next phase, during content enrichment, primary information on cultural activities can be extended with more substantive and facilitating information. Examples of facilitating information are 'how the weather will be' or 'how to get to the event with public transport'. Substantive information may include background information on the artist, press articles, images, etc.

A lot of event information is distributed by various content providers all over the Internet. (The Flemish content providers who cooperated in this project are *CultuurNet Vlaanderen* and *Vooruit*). They publish their events on blogs, web sites, calendars, social networks, etc. using different formats like simple HTML[2], iCalendar[3] or RSS[4]. As a result, the main challenge of content aggregation in the context of events is to provide a framework that supports the aggregation of data from heterogeneous data sources which use different standards to publish their events.

To deal with these different incoming data formats, a transformation is required to convert the event data from their original format to the internal event ontology, discussed in Section 3. For this purpose we use an XSL transformations engine. XSLT[5] is a declarative, XML-based language used for the transformation of XML documents into other XML documents. To support the aggregation of a new data format in the future, the XSL style sheet has to be extended with the mapping of this new format to the internal event ontology. A double detection mechanism eliminates redundant data: events with same date and the same location as existing events are considered as duplicate and will not be inserted into the database.

Enriching the events is realized by first extracting all the concepts out of the event description using linguistic processing. For this, we utilize *OpenCalais* and *i.Know's Information Forensics*[6] service. Afterwards, the extracted named entities are used for querying other data sets and incorporating the retrieved information. The extracted locations are employed for querying *Toerisme Vlaanderen*, *GeoNames* and *DBpedia* [7]. *Toerisme Vlaanderen* is a Flemish tourism information point with a lot of information on Flemish cities and regions, restaurants, hotels, and events. The extracted organisations and persons are utilized for querying the *DBpedia* data set and *BibNet*'s data set. *BibNet*[8] is a network of Flemish public libraries, which disseminates bibliographic information. The retrieved links are stored and offered as additional information sources to the end-user. By enriching the event information, the end-users get a lot of extra data to make an informed choice. Together with the recommendations, this will help them to evaluate and select the events which match with their personal preferences. Another advantage of this enrichment process is the semantic linking of event information to other data sets which incorporates the event information into the 'Web of Data'.

3 EVENT MODEL

To acquire a standardised procedure for the dissemination of events and to enable content-based recommendation techniques, a semantic metadata model for storing and exchanging event information is required. To represent the events as structured data, various standards exist. The most commonly used standard for describing events is iCalendar. This standard, which describes events for personal management purposes, defines an event as anything with a scheduled amount of time on a calendar. Although, the iCalendar format is able to describe a socio-cultural event, it can not express relations between the events. For instance, a festival can consist of many smaller events, i.e. concerts or music performances, which have to be related.

In the Linked Open Data community, the most used event model is the Music Ontology Events Model (Centre for Digital Music - University of London, 2007). AudioScrobbler (LinkingOpenData (W3C SWEO Community Project) - Centre for Digital Music, 2007) of Last.FM utilizes this model to describe events which are then linked to a user pro-

[2] http://tools.ietf.org/html/rfc2854
[3] http://tools.ietf.org/html/rfc5545
[4] http://www.rss-specifications.com
[5] http://www.w3.org/TR/xsltis
[6] http://www.opencalais.com, http://www.iknow.be
[7] http://www.toerismevlaanderen.be, http://www.geonames.org, http://dbpedia.org
[8] http://www.bibnet.be

file, for which FOAF[9] is used. This way, an event is modelled whenever a song is played. Originally, this model was intended to describe musical events, but due to its simplicity and usability, it has been proven useful in a wide range of contexts. This model describes an event as anything that has a spatial and temporal dimension. Such an event is described by its participating agents, its passive factors influencing the event, its products as a consequence of the event and a location in time as well as space. In addition, this model allows describing relations between events. However, its simplicity is also a disadvantage since the model lacks some advanced features, like pricing information, more detailed relations between events, minimum age for participation, etc., which are essential for describing events in our context.

Finally, the format that we adopted for modelling events is the EventsML-G2 (International Press Telecommunications Council, 2009) standard. This is a standard of the International Press Telecommunications Council[10] (IPTC) for conveying event information in a news industry environment. EventsML-G2 is intended for receiving, storing and exchanging event information from organisers as well as publishing event information by news providers. This model delivers the right context for our event descriptions since it allows describing events in different languages, together with their relations, the pricing information, the minimum age, etc.

Currently, the EventsML-G2 standard is described as an XML (Bray et al., 2006) schema. In order to apply semantic web techniques, we developed an OWL (McGuinness and van Harmelen, 2004) ontology of this standard. An alternative to make the EventsML-G2 schema usable in the Semantic Web Stack, is describing the schema in RDFS (Brickley, 2004). However, since OWL is a richer language for developing models and permits to mix RDFS and OWL constructs, we opted for describing the EventsML-G2 semantic ontology by using OWL. This OWL ontology is published online on the website of the Multimedia Lab research group of Ghent University[11].

By describing all the aggregated events using this ontology, a common format is created for exposing the event information. This common format acts as a unifying layer, relating all the information coming from different data providers. By providing a semantic ontology for these events, content-based recommendation algorithms are able to analyse the events in detail, and additionally, semantic web techniques can be used. Moreover, this semantic ontology allows complex queries for events by a SPARQL (Prud'hommeaux and Seaborne, 2007) endpoint and the enrichment of event descriptions with information from other data providers as described in Section 2.

EventsML-G2 has two manners for conveying event information: as a *conceptItem* or as a *knowledgeItem*. A *conceptItem* is aimed at describing an event solely. A *knowledgeItem* is intended for bundling a set of events which are managed as a whole. Given our context, i.e. publishing and recommending events from event organisers, we utilize the *conceptItem* to model the events. For interoperability issues, we modelled agents involved in an event, e.g. organisers or participants, using the FOAF ontology. This allows us to incorporate more easily information about the agents from other data sets afterwards. This additional advantage is the main difference between our developed ontology and the EventsML-G2 specification.

The descriptive metadata about the event is captured by the *Concept* class, which stores a description of the event, the title of the event, the event details, relations to other events, and the language of the event description. The relations to other events can be described by *owl:sameAs* for stipulating that this event is the same as another event, and by *skos:related*, *skos:broader* or *skos:narrower* for pinpointing the relations to other events. The *EventDetails* class contains a more complete event description and is linked to ten other classes for storing detailed information about the event, namely: *Dates, Location, ContactInfo, Agent, Language, Subject, OccurStatus, Registration, ParticipationRequirement,* and *Media*.

4 EVENT RECOMMENDATION

To handle the overload of information, a recommender system is necessary to assist users in finding the most relevant events. Traditionally, recommender systems have been categorized into two main classes: collaborative filtering (CF) and content-based (CB) methods. CF techniques are based on the hypothesis that a good method to find interesting content is to search for other people who have similar interests, and then recommend items that those similar users like (Breese et al., 1998). Most existing recommender applications utilize these CF techniques in making predictions about which items an e-service user is likely to be interested in (Linden et al., 2003). Traditional CF techniques, however, can not cope at all with time specific items, like events, which typically receive their ratings only after they have finished (Cornelis et al., 2005). Content-based (CB)

[9] http://xmlns.com/foaf/spec/

[10] http://www.iptc.org

[11] http://multimedialab.elis.ugent.be/ontologies/EventsML-G2/v1.0/

algorithms, which consider the internal structure of items and recommend items similar to those a user liked in the past, or hybrid algorithms, which are a combination of CF and CB techniques, can make up for these timely characteristics of events.

Although the details of various systems differ, CB recommender systems share in common a need for a description of the items that may be recommended (Pazzani and Billsus, 2007). Many CB recommender systems use relatively simple keyword extraction techniques to obtain characteristic properties from these descriptions. Subsequently, straightforward retrieval models, such as keyword matching, are utilized to generated recommendations (Bogers and van den Bosch, 2007). However, these techniques are known to have been outperformed by newer techniques that incorporate semantic knowledge structures, such as ontologies, to provide valuable domain knowledge for CB recommender systems (Middleton et al., 2003). Common-sense and domain-specific knowledge may be useful to give some meaning to the content of items, thus helping to generate more informative features than 'plain' attributes.

Knowledge sources, such as structured semantic information about events, can be brought to bear in determining similarities among events. The integration of semantic similarities for items allows the system to make inferences based on the underlying reasons for which a user may or may not be interested in a particular event (Mobasher et al., 2003). Experimental results in literature demonstrate that the integration of these ontological knowledge sources yields significant advantages in terms of learning user profiles and recommendation accuracy in contrast to the classical use of endogenous knowledge (extracted from the items themselves) (Semeraro et al., 2009).

However, acquiring such knowledge sources or semantically structured content is not a trivial task. Moreover, content (and especially event information) from various sources is mostly annotated using a multitude of different metadata formats, leading to serious comparison difficulties for the recommender. By converting the aggregated events to our proposed event model, we provide a uniform event structure that can be utilized for semantic analysis. By enriching the event description based on a set of knowledge sources, we generate a memory of semantic competencies that can be exploited to reason about the content as well as to support the user profiling and recommendation process. Moreover, based on our service-oriented bus-architecture as described in Section 5, we can easily integrate any existing (ontology-based) recommendation algorithm in our event distribution system.

To efficiently calculate the personal suggestions based on the adopted (ontology-based) algorithm for a potentially high number of users, we designed the recommendation framework which is illustrated in Figure 1. To meet the high requirements of a large recommendation system, we opted for a highly scalable approach to store and process user and event metadata completed with online-generated user interaction behaviour (i.e. user feedback on the events). Moreover, due to the computational burden of most recommendation algorithms, the calculation of personal suggestions has to be realized as a distributed computation task.

The proposed recommender framework consists of two webservices and three small, independent applications working together to collect user behaviour and metadata content from participating web sites and to generate high quality recommendations for their visitors. The webservices act as a communication channel between the recommendation system and the other parts of the event distribution platform (i.e. client websites or other webservices). The *Data Insertion webservice* deals with the processing of new users, events and feedback on events, which are fed into the recommender. Based on these data, personal event recommendations are generated which can be queried through the *Recommendation webservice*.

Each of the three applications accomplishes a single well-defined task. The first application, the *Data Inserter*, inserts users, events and user feedback into the storage system. Secondly, the *Calculator* pulls these data from the storage system and generates personal suggestions based on the implemented recommendation algorithm. Next, these suggestions are stored in the *Recommendation Cache*, making them immediately available for retrieval by the *Recommendation webservice*. Depending on the work load, several instances of this *Data Inserter* and *Calculator* may be created. At regular intervals, the third application, i.e. the *Scheduler*, generates a calculation task which is split into disjunctive jobs for the *Calculator* instances. The results of these calculation jobs are merged and inserted into the *Recommendation Cache* as an update of the personal suggestions.

Because of performance reasons, the different applications are not in direct contact with each other, but use a queue service for inter-process communication. Moreover, the storage system is optimized for efficiently serving two commonly-used data requests: on the one hand inserting singular user, event or feedback information originating from the *Data Insertion webservice*, and on the other hand, querying massive feedback and event information to feed the instances of the *Calculator* application.

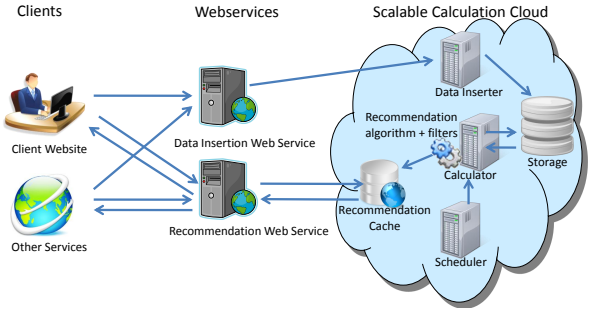

Figure 1: The recommendation framework.

Besides the storage and calculation functionality, the recommendation framework consists of various contextual post-filters which take into account the personal event selection criteria. These filters operate on the list of personal event recommendations and can eliminate the suggestions which do not conform to the personal event selection criteria. The personal event selection criteria, which can explicitly be specified by the end-user, are personal requirements regarding the price, language, location, participation requirements or date of the event. E.g., events that are located too far from the hometown of the user are removed from the suggestions list, since they are meaningless for the end-user. Accordingly, suggestions which completely fulfil all the personal requirements are favoured during the generation of the recommendation list. Through the proposed event ontology, as discussed in Section 3, and the enrichment process, as explained in Section 2 event suggestions can easily be verified against the stated event requirements before distribution.

5 ARCHITECTURE

Figure 2 gives a schematic overview of the infrastructure of the complete event distribution platform, a loosely coupled bus-architecture consisting of 4 components: aggregation, enrichment, recommendation, and distribution. This architecture supports the flow of personalized, enriched cultural activities and connects the various functional components of our system. An enterprise service bus (ESB) provides an open, standards-based connectivity infrastructure for this Service-Oriented Architecture (SOA)[12]. The SOA enables flexible connectivity of applications or resources by representing every application or resource as a service with a standardized interface. This flexibility allows new and existing applications to be easily and quickly combined to address changing business needs.

In the context of the event distribution system, this means that the different components that compose the general architecture are modelled as separate services where the SOA allows them to exchange data with one another as they participate in business processes. The main advantage of this SOA is to obtain a loose coupling between the different modules of the system in which services can be very heterogeneous. The implementation of these services is language and technology independent; only communications must be agreed on. The high flexibility supported by this technology allows an easy integration of new sources (e.g. aggregation sources or distribution targets) and remove or modify existing sources without interrupting the working system. In addition, using a standardized technology provides also the advantage of reusable existing middleware services. Existing implementations contain many pre-built components which can be reused with a minimal effort. Existing components are available in the domain of logic and orchestration, databases and data manipulation, basic interfacing, security, logging, etc. Although an advanced SOA setup requires some additional start-up costs and a slight increase of complexity, it provides the freedom to expand and modify the active setup at any time while requiring only a small amount of effort.

The ESB technology we used for this project is the Open Enterprise Service Bus[13]. OpenESB is an open-source project with the aim of building an Enterprise Service Bus that provides a flexible and extensible platform on which to build SOA and application integration solutions. Pulling event information from the web is triggered by a scheduler. On predefined times, the scheduler sends a message to the input services to check the input sources for new content. Orchestration between services is handled by a workflow engine, which is integrated in the service bus architecture and supports the execution of WS-BPEL processes (Business Process Execution Language for Web Services[14]). WS-BPEL is built on top of XML specifications and provides a language for the specification of business processes and business interaction protocols. An executable WS-BPEL process is defined by a control flow that consists of a combination of basic and structured activities. WS-BPEL (and SOA in general) supports the Simple Object Access Protocol (SOAP), which is a specification for exchanging structured information between services.

[12] http://opengroup.org/projects/soa/

[13] https://open-esb.dev.java.net/

[14] http://docs.oasis-open.org/wsbpel/

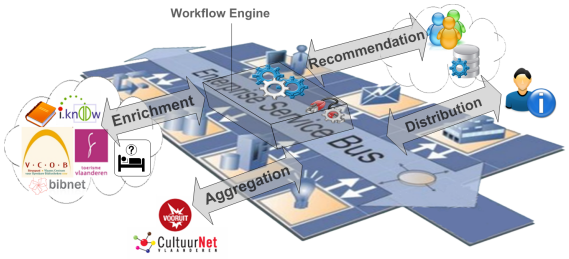

Figure 2: The service bus architecture.

6 CONCLUSIONS

Online event-data from various providers may emerge as a dynamic mix of many different types of resources and facilities. This multiplicity of technologies and metadata descriptions requires a uniform event model and dynamic event distribution architecture. In this paper, we proposed an event model based on the EventsML-G2 standard and a service-oriented bus architecture for event recommender and distribution systems. Aggregation, enrichment, recommendation and distribution are supported through services, which can be created and modified dynamically to suit the current needs. This physical loose-coupling provides scalability advantages such as high-availability, fault-tolerance, and load balancing. The combination of the semantic event model and service-oriented architecture allows the application of an exchangeable content-based recommendation algorithm. Enriching the event metadata enables the incorporation of semantic knowledge structures into the recommendation algorithm.

ACKNOWLEDGEMENTS

The research activities that have been described in this paper were funded by Ghent University, K.U. Leuven, VUB, VRT-medialab, Interdisciplinary Institute for Broadband Technology (IBBT) through the CUPID project (50% co-funded by industrial partners), the Institute for the Promotion of Innovation by Science and Technology in Flanders (IWT), the Fund for Scientific Research-Flanders (FWO-Flanders), and the European Union.

REFERENCES

Bogers, T. and van den Bosch, A. (2007). Comparing and evaluating information retrieval algorithms for news recommendation. In *RecSys '07: Proceedings of the 2007 ACM conference on Recommender systems*, pages 141–144, New York, NY, USA. ACM.

Bray, T., Paoli, J., Sperberg-McQueen, C., Maler, E., and Yergeau, F., editors (2006). *Extensible Markup Language (XML) 1.0 (Fourth Edition)*. W3C Recommendation. World Wide Web Consortium. Available at `http://www.w3.org/TR/2006/REC-xml-20060816/`.

Breese, J., Heckerman, D., and Kadie, C. (1998). Empirical Analysis of Predictive Algorithms for Collaborative Filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pages 43–52, Madison, USA.

Brickley, D., editor (2004). *RDF Vocabulary Description Language 1.0: RDF Schema*. W3C Recommendation. World Wide Web Consortium. Available at `http://www.w3.org/TR/rdf-schema/`.

Centre for Digital Music - University of London (2007). The Event Ontology. Available at `http://purl.org/NET/c4dm/event.owl`.

Cornelis, C., Guo, X., Lu, J., and Zhang, G. (2005). A Fuzzy Relational Approach to Event Recommendation. In *Proceedings of the 1st Indian International Conference on Artificial Intelligence*, pages 2231–2242, Pune, India.

International Press Telecommunications Council (2009). EventsML-G2 Specification – Version 1.1. Available at `http://www.iptc.com/std/EventsML-G2/EventsML-G2\_1.1.zip`.

Linden, G., Smith, B., and York, J. (2003). Amazon.com Recommendations: Item-to-item Collaborative Filtering. *IEEE Internet Computing*, 7(1):76–80.

LinkingOpenData (W3C SWEO Community Project) - Centre for Digital Music (2007). Audioscrobbler RDF Service. Available at `http://dbtune.org/last-fm/`.

McGuinness, D. and van Harmelen, F., editors (2004). *OWL Web Ontology Language: Overview*. W3C Recommendation. World Wide Web Consortium. Available at `http://www.w3.org/TR/owl-features/`.

Middleton, S. E., Shadbolt, N., and Roure, D. D. (2003). Ontology-based recommender systems.

Mobasher, B., Jin, X., and Zhou, Y. (2003). Semantically enhanced collaborative filtering on the web. In *Proceedings of the 1st European Web Mining Forum (EWMF2003)*, pages 57–76.

Pazzani, M. and Billsus, D. (2007). *Content-Based Recommendation Systems*. Springer.

Prud'hommeaux, E. and Seaborne, A., editors (2007). *SPARQL Query Language for RDF*. W3C Recommendation. World Wide Web Consortium. Available at `http://www.w3.org/TR/rdf-sparql-query/`.

Semeraro, G., Lops, P., Basile, P., and de Gemmis, M. (2009). Knowledge infusion into content-based recommender systems. In *RecSys '09: Proceedings of the third ACM conference on Recommender systems*, pages 301–304, New York, NY, USA. ACM.